

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

Priya is analyzing encrypted messages in a research project. She wants to analyze the frequency of each character in a given paragraph. The characters should be stored in a TreeMap so that the output is sorted in ascending order of characters automatically.

You are required to build a Java program that:

Uses a `TreeMap<Character, Integer>` to count how many times each character appears in the message. Ignores spaces and considers only alphabets (case-sensitive). Outputs the frequencies of characters in sorted order.

You must use a TreeMap in the class named MessageAnalyzer.

Input Format

The first line of input contains an integer n, the number of lines in the message.

The next n lines each contain a string (the encrypted message line).

Output Format

The first line of output prints: "Character Frequency:"

Then print each character and its frequency in the format: "<character>: <count>"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
Hello World
Java

Output: Character Frequency:

H: 1

J: 1

W: 1

a: 2

d: 1

e: 1

l: 3

o: 2

r: 1

v: 1

Answer

```
// You are using Java
import java.util.Scanner;
import java.util.TreeMap;
```

```
class MessageAnalyzer {
    private TreeMap<Character, Integer> map = new TreeMap<>();
```

```
    public void analyzeMessage(String message) {
        for (char ch : message.toCharArray()) {
            if (ch != ' ' && Character.isAlphabetic(ch)) {
                map.put(ch, map.getOrDefault(ch, 0) + 1);
```

```

    }
    }
}

public void displayFrequency() {
    System.out.println("Character Frequency:");
    for (char ch : map.keySet()) {
        System.out.println(ch + ": " + map.get(ch));
    }
}
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());

        MessageAnalyzer analyzer = new MessageAnalyzer();

        for (int i = 0; i < n; i++) {
            String line = sc.nextLine();
            analyzer.analyzeMessage(line);
        }

        analyzer.displayFrequency();
        sc.close();
    }
}

```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

John is organizing a fruit festival, and the quantities of various fruits are stored in a HashMap where fruit names are keys and quantities are values.

Help him develop a program to find the total quantity of fruits for the festival by summing up the values in the HashMap.

Input Format

The input consists of fruit quantities in the format 'fruitName:quantity', where fruitName is the name of the fruit(a string), and quantity is a double value representing the quantity.

The input is terminated by entering "done".

Output Format

The output prints a double value, representing the sum of values in the HashMap, rounded off to two decimal places.

If the value is not numeric, print "Invalid input".

If any special characters other than ':' are entered, print "Invalid format".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Banana:15.2

Orange:56.3

Mango:47.3

done

Output: 118.80

Answer

// You are using Java

```
import java.util.HashMap;
```

```
import java.util.Scanner;
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        HashMap<String, Double> map = new HashMap<>();
```

```
        double sum = 0.0;
```

```
        boolean invalidInput = false;
```

```
        boolean invalidFormat = false;
```

```
        while (true) {
```

```
            String input = sc.nextLine();
```

```
            if (input.equals("done")) {
```

```
                break;
```

```
            }
```

```
            // Check invalid format: must contain only one ":" and no other special  
            char
```

```
            if (!input.contains(":") || input.indexOf(":") != input.lastIndexOf(":")) {
```

```
        invalidFormat = true;
        break;
    }
```

```
String[] parts = input.split(":");
```

```
if (parts.length != 2) {
    invalidFormat = true;
    break;
}
```

```
String fruit = parts[0];
String value = parts[1];
```

```
// Check if fruit name or value contains disallowed characters
if (!fruit.matches("[A-Za-z]+") || !value.matches("[0-9.]+")) {
    invalidInput = true;
    break;
}
```

```
try {
    double qty = Double.parseDouble(value);
    map.put(fruit, qty);
} catch (Exception e) {
    invalidInput = true;
    break;
}
}
```

```
if (invalidFormat) {
    System.out.println("Invalid format");
} else if (invalidInput) {
    System.out.println("Invalid input");
} else {
    for (double val : map.values()) {
        sum += val;
    }
    System.out.printf("%.2f", sum);
}
sc.close();
}
```

}

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

A city traffic management system needs to track vehicles entering a toll booth. Each vehicle is uniquely identified by its registration number. The system should allow adding vehicles to a record, ensuring that no duplicate registration numbers exist. The vehicles should be stored in a HashSet, which does not guarantee any specific order.

Your task is to implement a program using a HashSet that allows adding vehicle details and displaying the records.

Input Format

The first line of input contains an integer N - the number of vehicles.

The next N lines contain details of each vehicle in the format: "RegNumber

OwnerName VehicleType"

1. RegNumber (String) - A unique registration number (Alphanumeric).
2. OwnerName (String) - The name of the vehicle owner.
3. VehicleType (String, Car, Bike, or Truck) - The type of vehicle.

If a vehicle with the same registration number is already present, ignore the duplicate entry.

Output Format

The output prints the unique vehicle records in any order (since HashSet does not maintain order).

Output format: "RegNumber OwnerName VehicleType"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

KA01AB1234 John Car

MH02CD5678 Alice Bike

DL03EF9012 Bob Truck

TN04GH3456 Mike Car

KA01AB1234 John Car

Output: TN04GH3456 Mike Car

KA01AB1234 John Car

MH02CD5678 Alice Bike

DL03EF9012 Bob Truck

Answer

```
// You are using Java
import java.util.HashSet;
import java.util.Scanner;
```

```
class Vehicle {
    String regNumber;
    String ownerName;
    String vehicleType;
```

```
public Vehicle(String regNumber, String ownerName, String vehicleType) {  
    this.regNumber = regNumber;  
    this.ownerName = ownerName;  
    this.vehicleType = vehicleType;  
}
```

```
@Override  
public int hashCode() {  
    return regNumber.hashCode();  
}
```

```
@Override  
public boolean equals(Object obj) {  
    if (this == obj) return true;  
    if (obj == null || getClass() != obj.getClass()) return false;  
    Vehicle v = (Vehicle) obj;  
    return regNumber.equals(v.regNumber);  
}
```

```
@Override  
public String toString() {  
    return regNumber + " " + ownerName + " " + vehicleType;  
}  
}
```

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int N = sc.nextInt();  
        sc.nextLine();  
  
        HashSet<Vehicle> set = new HashSet<>();  
  
        for (int i = 0; i < N; i++) {  
            String input = sc.nextLine();  
            String[] parts = input.split(" ");  
  
            String reg = parts[0];  
            String owner = parts[1];  
            String type = parts[2];  
  
            set.add(new Vehicle(reg, owner, type));  
        }  
    }  
}
```

```
}  
    for (Vehicle v : set) {  
        System.out.println(v);  
    }  
  
    sc.close();  
}  
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Assist Pranitha in developing a program that takes an integer N as input, representing the number of names to be read. Then read N names and store them in an ArrayList. Finally, input a search string and output the frequency of that string in the list of names.

Note: Some parts of the code are provided as snippets, and you need to complete the remaining sections by writing the necessary code.

Input Format

The first line of input consists of an integer N, representing the number of names to be read.

The following N lines consist of N names, as a string.

The last line consists of a string, representing the name to be searched.

Output Format

The output prints a single integer, representing the frequency of the specified name in the given list.

If the specified name is not found, print 0.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

Alice

Bob

Ankit

Alice

Pranitha

Alice

Output: 2

Answer

// You are using Java

```
import java.util.ArrayList;
```

```
import java.util.Scanner;
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        int N = sc.nextInt();
```

```
        sc.nextLine(); // consume newline
```

```
        ArrayList<String> names = new ArrayList<>();
```

```
        for (int i = 0; i < N; i++) {
```

```
            String name = sc.nextLine();
```

```
            names.add(name);
```

```
        }
```

```
String search = sc.nextLine();
int count = 0;

for (String name : names) {
    if (name.equals(search)) {
        count++;
    }
}

System.out.println(count);

sc.close();
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Vikram loves listening to music and wants to create a simple playlist manager using Java Collections. The playlist supports the following operations:

"ADD <song>" Adds the song to the end of the playlist. "REMOVE <song>" Removes the first occurrence of the song from the playlist. If the song is not found, do nothing. "SHOW" Displays all songs in the playlist in order. If the playlist is empty, print "EMPTY". "NEXT" Moves to the next song in the playlist and prints its name. If the playlist is empty, print "EMPTY".

The playlist maintains a "current song" position that starts at the first song when it's added. The NEXT command moves to the next song and prints it, wrapping around to the first song after reaching the last song. When removing songs, the current position adjusts accordingly to maintain

proper navigation.

Help Vikram implement this playlist manager.

Input Format

The first line of the input consists of an integer n, the number of operations.

The next n lines, each containing a command:

- "ADD <song>"
- "REMOVE <song>"
- "SHOW"
- "NEXT"

Output Format

For each "SHOW" command, print the songs in order, separated by spaces.

For each "NEXT" command, print the next song in the playlist.

If no song exists, print "EMPTY".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7

ADD song1

ADD song2

SHOW

NEXT

REMOVE song2

SHOW

NEXT

Output: song1 song2

song2

song1

song1

Answer


```
import java.util.LinkedList;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        int n = Integer.parseInt(scanner.nextLine());

        LinkedList<String> playlist = new LinkedList<>();
        int currentIndex = -1;

        for (int i = 0; i < n; i++) {
            String command = scanner.nextLine();

            if (command.startsWith("ADD ")) {
                String song = command.substring(4);
                playlist.add(song);

                if (currentIndex == -1) {
                    currentIndex = 0;
                }

            } else if (command.startsWith("REMOVE ")) {
                String song = command.substring(7);
                if (!playlist.isEmpty()) {
                    int removeIndex = playlist.indexOf(song);
                    if (removeIndex != -1) {
                        playlist.remove(song);

                        if (playlist.isEmpty()) {
                            currentIndex = -1;
                        } else if (removeIndex < currentIndex) {
                            currentIndex--;
                        } else if (removeIndex == currentIndex) {
                            if (currentIndex == playlist.size()) {
                                currentIndex = 0;
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```
} else if (command.equals("SHOW")) {  
    if (playlist.isEmpty()) {  
        System.out.println("EMPTY");  
    } else {  
        for (String s : playlist) {  
            System.out.print(s + " ");  
        }  
        System.out.println();  
    }  
  
    } else if (command.equals("NEXT")) {  
        if (playlist.isEmpty()) {  
            System.out.println("EMPTY");  
        } else {  
            currentIndex = (currentIndex + 1) % playlist.size();  
            System.out.println(playlist.get(currentIndex));  
        }  
    }  
}  
  
scanner.close();  
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Rohan Chelliah.Sk
Email: 241801230@rajalakshmi.edu.in
Roll no: 241801230
Phone: 6374135924
Branch: REC
Department: AI & DS - Section 3
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 9_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bobby is tasked with processing a sequence of numbers from a monitoring system. He needs to extract a strictly increasing subsequence using an ArrayList. The program should dynamically add numbers to the ArrayList only if they are greater than the last number currently stored in the list. Bobby aims to efficiently utilize the dynamic resizing and indexing features of the ArrayList to solve this problem.

Help Bobby implement this solution.

Input Format

The first line of input consists of an integer N, representing the number of elements.

The second line consists of N space-separated integers, representing the elements.

Output Format

The output prints the list of integers in increasing sequence, ignoring out-of-order elements.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 7

3 5 9 1 11 7 13

Output: [3, 5, 9, 11, 13]

Answer

```
// You are using Java
import java.util.ArrayList;
import java.util.Scanner;
```

```
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int N = scanner.nextInt();
        ArrayList<Integer> list = new ArrayList<>();

        for (int i = 0; i < N; i++) {
            int num = scanner.nextInt();

            if (list.isEmpty() || num > list.get(list.size() - 1)) {
                list.add(num);
            }
        }

        System.out.println(list);
        scanner.close();
    }
}
```

Status : Correct

Marks : 10/10